Release Notes for the Freescale P4080DS QNX Neutrino 6.5.0 BSP Revision 1.0.0

Date of this Edition: Aug 30, 2011

1. System Requirements

Target Requirements

1. QNX Neutrino RTOS 6.5.0
    - **Note: This BSP requires an updated procnto-booke-smp with the fixes for QNX Ref# 92445 'P4080DS: procnto hang on smp startup with 36-bit IMMR value' and QNX Ref# 90382 'P4080DS: netperf error 'netperf: data send error: Socket operation on non-socket''. The updated procnto-booke-smp is included with this BSP.**
2. U-Boot 2010.6 bootloader from the Freescale Linux SDK 2.2
3. DPAA FM Microcode 101.6.0
4. Board version: P4080DS with a P4080 rev2 processor
5. 2-4GB DDR SDRAM (interleaved or non-interleaved)
6. 128 MB NOR flash

Host Requirements

1. QNX Momentics 6.5.0
2. Terminal emulation program (Qtalk, Momentics IDE Terminal, tip, HyperTerminal, etc.)
3. RS-232 serial port and serial cable, or a USB-to-serial cable
4. Ethernet link

Note:
Throughout this document, you may see *reference numbers* associated with particular issues, changes, etc. When corresponding with QNX Technical Support staff about a given issue, please quote the relevant reference number. You might also find the reference numbers useful for tracking issues as they become fixed.

For the most up-to-date version of these release notes, go to our website (www.qnx.com), log into your myQNX account, and then go to the Foundry27 Download area where this BSP was originally obtained.

For information on installing this BSP, see the installation notes at the Foundry27 Download area for this BSP.

Technical support

To obtain technical support for any QNX product, visit the Support + Services area on our website (www.qnx.com). You'll find a wide range of support options, including community forums.

2. System Layout

The P4080 has a 36-bit physical address space and with the 2010.6 U-Boot all device addresses and the IMMR/CCSR region are configured above 4Gb (memory starts at physical address 0). Within U-Boot commands the 36-bit addresses are mapped to 32-bits by clearing the upper 4-bits of this physical address which puts them in the upper 2Gb of the 4Gb 32-bit address space (so to access Flash within U-boot commands you use address 0xE800_0000 rather than 0xF_E800_0000). U-Boot does not map the upper 2Gb of SDRAM so there is no conflict between these mappings and SDRAM within U-Boot.

Virtual addresses are 32-bits with this processor and BSP. The 36-bit physical addresses are mapped to virtual addresses by the Neutrino kernel as needed.

Start	End	Item
0x0_0010_0000		OS Image Loaded
0x0_0000_0000	0x3_FFFF_FFFF	2-16Gb SDRAM
0xF_8000_0000	0xF_83FF_FFFF	PCIe1 Memory (slot 1)* Default**
0xF_8400_0000	0xF_87FF_FFFF	PCIe2 Memory (slot 3)* Default**
0xF_8800_0000	0xF_8BFF_FFFF	PCIe3 Memory (slot 2)* Default**
0xF_E000_0000	0xF_E7FF_FFFF	Promjet***
0xF_E800_0000	0xF_EFFF_FFFF	Nor Flash (on eLBC)
0xF_F400_0000	0xF_F41F_FFFF	Buffer Manager
0xF_F420_0000	0xF_F43F_FFFF	Queue Manager
0xF_F800_0000	0xF_F800_FFFF	PCIe1 IO (slot 1)*
0xF_F801_0000	0xF_F801_FFFF	PCIe2 IO (slot 3)*
0xF_F802_0000	0xF_F802_FFFF	PCIe3 IO (slot 2)*
0xF_F803_0000	0xF_F812_FFFF	NAND Flash Bank 1 (eLBC Chip Select 2)***
0xF_F813_0000	0xF_F822_FFFF	NAND Flash Bank 2 (eLBC Chip Select 4)***
0xF_F823_0000	0xF_F832_FFFF	NAND Flash Bank 3 (eLBC Chip Select 5)***
0xF_F833_0000	0xF_F842_FFFF	NAND Flash Bank 4 (eLBC Chip Select 6)***
0xF_FE00_0000	0xF_FEFF_FFFF	IMMR/CCSR Device Register Space
0xF_FFDF_0000	0xF_FFDF_0FFF	PIXIS Registers (eLBC Chip Select 3)

Note: * The PCIe bus numbers do not correspond to slot numbers. Slots 1, 2, and 3 generally correspond to PCIe hardware buses 1, 3, and 2 respectively but this depends on the RCW configuration.
Note: ** These correspond to the default 64Mb PCIe register memory window size. This can be set anywhere in the range 4-512Mb using the startup -W command option, which would change the mapping accordingly.
Note: *** The Promjet and NAND Flash Banks are not installed on the P4080DS board.
Note: These are physical addresses. Within U-Boot commands they are mapped down to 32-bits by clearing the upper 4 bits.

3. Getting Started

3.1 Building the BSP

You can build a BSP OS image from the source code or the binary components contained in a BSP package. For instructions about building a BSP OS image, please refer to the chapter Working with a BSP in the Building Embedded Systems manual.

3.2 Connect your Hardware

Connect the serial cable to the first serial port of the P4080DS board to the first serial port of your host machine. There are 2 serial ports on P4080DS. Use the one which is near the boundary of the board. Usually you should see some U-Boot output on the console when you connect cable to the correct port. If you have a Neutrino host with a serial mouse, you may have to move the mouse to the second serial port on your host, because some terminal programs require the first serial port.

The correct terminal settings of the program handling serial connection should be:

baudrate	115200
data	8 bit
parity	none
stop	1bit
flow control	none

3.3 Setup your environment

1. Power on your target. You should see the U-Boot output on your console.
2. Connect an ethernet cable to the RGMII PHY port available on the back side of the board, below the serial ports.

4. Boot the IFS image

You can use an SD/MMC memory card, TFTP download (the default) or serial download to transfer an OS image to the board, as described below.

4.1 Save U-Boot environment

The commands in the following procedures involve changes to the default U-Boot configuration and before you do this you should issue the following commands to preserve some of the factory U-Boot settings should you need them later:

```
=> setenv bootcmd_lnx $bootcmd
=> setenv hwconfig_orig $hwconfig
=> saveenv
Saving Environment to Flash...
Un-Protected 1 sectors
Erasing Flash...
flash erase done
Erased 1 sectors
Writing to Flash... done
Protected 1 sectors
=>
```

Remember to do this for each flash bank you intend to boot QNX Neutrino 6.5.0 from as each bank has its own U-Boot and therefore its own copy of the U-Boot environment variables.

The original hwconfig setting (`fsl_ddr:ctlr_intlv=cacheline,bank_intlv=cs0_cs1`) enables memory interleaving for memory DIMMs with 2 chip selects. Note that if you are using memory with 4 chip selects (e.g. 8G DIMMs) you must modify this setting to `fsl_ddr:ctlr_intlv=cacheline,bank_intlv=cs0_cs1_cs2_cs3` to have U-Boot correctly configure interleaving. Using the wrong setting will cause U-Boot and/or QNX to hang or behave strangely due to the mis-configured memory. If you are unsure of the number of banks for your DIMMs, boot with memory interleave disabled and QNX will report the number of banks to the console when adding the DIMMs to system memory. Then reconfigure the U-Boot hwconfig memory interleave setting with the correct number of bank chip selects.

4.2 Boot via SD/MMC memory card

This method requires that you put the raw image generated by BSP (by default at $BSP_ROOT/images/ifs-p4080ds.raw) on a DOS format SD/MMC card. As soon as U-Boot starts, press any key so that U-Boot stops and doesn't boot the prebuilt linux kernel. Configure U-Boot parameters as follows:

```
=> setenv bootfile ifs-p4080ds.raw
=> setenv loadaddr 0x100000
=> setenv bootcmd 'mmcinfo;fatload mmc 0:1 $loadaddr $bootfile;go $loadaddr'
=> setenv bootdelay 2
=> saveenv
Saving Environment to Flash...
Un-Protected 1 sectors
Erasing Flash...
flash erase done
Erased 1 sectors
Writing to Flash... done
Protected 1 sectors
=> boot
```

4.3 Boot via tftp

This method requires that you put the raw image generated by BSP (by default at $BSP_ROOT/images/ifs-p4080ds.raw) to a TFTP server. This server must be reachable via board and preferably should be on the same LAN. As soon as U-Boot starts, press any key so that U-Boot stops and doesn't boot the prebuilt linux kernel. Configure U-Boot parameters as follows:

```
=> setenv ipaddr 10.42.175.189
=> setenv serverip 10.42.175.188
=> setenv bootfile ifs-p4080ds.raw
=> setenv loadaddr 0x100000
=> setenv bootcmd 'tftpboot $loadaddr $bootfile; go $loadaddr'
=> setenv bootdelay 2
=> setenv ethprime FM1@DTSEC2
=> saveenv
Saving Environment to Flash...
Un-Protected 1 sectors
Erasing Flash...
flash erase done
Erased 1 sectors
Writing to Flash... done
```

```
Protected 1 sectors
=> boot
```

The U-Boot ethprime variable is set so that the RGMII network port on the mainboard is used (below the serial ports, next to the USB port), but any of the DTSEC ports or a PCIe e1000 network port can also be used. Some examples for setting ethprime are:

- FM1@DTSEC2 -- RGMII port on back of P4080DS
- FM2@DTSEC1 -- RGMII port on back of P4080DS (alternative RCW config)
- FM2@DTSEC3 -- 3rd port of SGMII 4x1G
- FM2@DTSEC4 -- 4th port of SGMII 4x1G
- e1000#0 -- PCIe e1000 NIC

The available interfaces are reported in the U-Boot log.

Note: U-Boot cannot boot from the 10G interfaces.

4.4 Boot via serial

This method requires an SREC image. You have to modify the build file to create this format. Change this:

[virtual=ppcbe,raw]

to this:

[virtual=ppcbe,srec]

Rebuild the image. On your target, type:

```
=>: setenv loads_echo 0
=>: saveenv
=>: loads
```

On your host, copy the image to the serial port that's connected to the board. For example, on a Neutrino host: cp ifs-p4080ds.srec /dev/ser1 On a Windows host, you can use Hyperterminal's transfer feature to copy the image as a text file.

```
## First Load Addr = 0x00100000
## Last  Load Addr = 0x0023955B
## Total Size      = 0x0013955C = 1283420 Bytes
## Start Addr      = 0x00101E38
=>:
```

Type **`go start_addr`**

4.5 Booting Neutrino

At this point, you should see output similar to this when U-Boot finishes downloading and booting the IFS image:

```
## Starting application at 0x00100000 ...
   .
   .
   .
Welcome to QNX Neutrino 6.5.0 on the Freescale P4080DS Board
#
```

Congratulations, the QNX 6.5.0 kernel is running on your system! You can test the OS simply by executing any shell builtin command or any command residing within the OS image (e.g. ls). Once the initial image is running, you can update the OS image using the network and flash drivers. For sample command lines, please see the

"Driver Command Summary" section.

5. Writing the IFS image to flash using the boot loader

6. Creating a flash partition

7. Driver Command Summary

Component	Buildfile Command	Required Binaries	Required Libraries	Source Location
Startup	startup-p4080ds -c0xFFE000000 -t100000000 -v	startup-p4080ds	libstartup.a	src/hardware/startup/boards/p4080ds
Serial	devc-ser8250 -e -b115200 0xffe11c500,14 waitfor /dev/ser1 reopen /dev/ser1 devc-ser8250 -e -b115200 0xffe11c600,14 waitfor /dev/ser2	devc-ser8250	none	src/hardware/devc
USB	io-usb -t memory=/memory/below4G -d ehci-p2020 ioport=0xfFE210100,irq=12,memory=/memory/below4G waitfor /dev/io-usb/io-usb 10 devb-umass cam pnp mem name=below4G	devu-ehci-p2020.so	libusbdi.so io-blk.so io-usb usb devb-umass libcam.so fs-dos.so fs-qnx4.so fs-ext2.so cam-disk.so cam-cdrom.so	"prebuiltonly"
I2C	i2c-mpc8572 -i16 -p0xffe118000 --u0 (for controller 1) waitfor /dev/i2c0 i2c-mpc8572 -i16 -p0xffe118100 --u1 (for controller 2) waitfor /dev/i2c1 i2c-mpc8572 -i17 -p0xffe119100 --u3 (for controller 4) waitfor /dev/i2c3	i2c-mpc8572	libi2c-master.a	src/hardware/i2c/mpc8572
PCI	pci-p4080	pci-p4080	none	/src/hardware/pci
SPI	spi-master -d ppc_espi	spi-master	spi-ppc_espi.so	/src/hardware/spi/master /src/hardware/spi/ppc_espi
NOR Flash	devf-generic -s 0xFE8000000,128M,,,128k,2,1 -r	devf-generic flashctl	libmtd-flash.a	/src/hardware/flash
RTC	rtc –v ds3232 /dev/i2c1 rtc -s -v ds3232 /dev/i2c1	rtc date	libutil.a	/src/utils/r/rtc
Network	io-pkt-v4-hc -ptcpip stacksize=65536 -dp40xx mac=00049F02XXX0 -de1000 OR /bin/netsetup auto mac=00049F02XXX0	io-pkt-v4-hc	devnp-p40xx.so libdpaaS.a libNetCommSwS.a devnp-e1000.so	/src/hardware/devnp /src/hardware/devnp/lib/dpaa /src/lib/3rdparty/netcomm devnp-e1000 "prebuiltonly"

PCI

run **pci-p4080** to enable PCI functionality on your board. PCI express controller selection depends upon the RCW settings. Make sure that the RCW settings enable the PCIe bus on the particular slot you are using for a PCIe device. To use a PCI-express device on P4080DS, the device should be connected to the slot before PCI server is run. To determine whether the connected device is detected successfully or not, run following command line utility

```
pci -v
```

This shall output all PCI devices currently detected on your system.

Depending on the RCW configuration, you can connect PCI express devices to the lower 2 or 3 slots on the board, labeled "PCI EXPRESS SLOT #1", "PCI EXPRESS SLOT #2", and "PCI EXPRESS / SGMII SLOT #3". The first two slots are dedicated to PCIe, the third can be PCIe or a 4-port 1Gb SGMII network card depending on the RCW settings.

The PCIe Memory window 36-bit CPU address ranges are mapped to a 32-bit range in the PCIe Memory address space to allow for devices with 32-bit BAR registers. This is done by truncating the 36-bit CPU address to 32-bits (for example 0xF_8000_0000 would become 0x0_8000_0000). The resulting address ranges overlap with

the SDRAM addresses on the PCIe bus and therefore cannot be used for DMA. To avoid conflicts with PCIe DMA the SDRAM in these address ranges are reserved by the startup. To control the memory lost to these conflicts the `startup-p4080` command includes a `-W` option to change the default size of these PCIe memory windows from the 64Mb to anywhere in the range of 4Mb to 512Mb. For example to reduce the PCIe Memory windows to 32Mb, change the default startup command to:

```
startup-p4080ds -c0xFFE000000 -t100000000 -v -W32
```

Note that this will reduce the PCIe Memory space available to map the device registers for devices on each PCIe bus, so make sure the chosen size is large enough for any PCIe devices you intend to support as well as for any PCI bridges on the bus. Only PCIe buses enabled in the RCW and by U-Boot have this SDRAM reserved, so a system with 2 PCIe buses will only lose 2 x 64Mb of SDRAM in the default configuration.

RTC

The RTC used in P4080DS board is a DS3232. Make sure the i2c driver is up and running before running the RTC utility, as the RTC DS3232 is connected to the second i2c bus.

If the i2c driver is not running, issue the following command

```
i2c-mpc8572 -i16 -p0xffe118100 --u1
```

To run the RTC utility, use the following command:

```
rtc –v ds3232 /dev/i2c1
```

This command updates the current time and date from the hardware clock in the board.

To set the hardware clock in the board with the current date and time, use the following command:

```
rtc –s –v ds3232 /dev/i2c1
```

NOR Flash

Run

```
devf-generic -s 0xFE8000000,128M,,,128k,2,1 –r
```

to run the generic Flash filesystem driver on your board. After running this command, two partitions will get created. Normally the file names are as below:

/dev/fs0 which is the default mount point for socket 0
/dev/fs0p0 which has the raw access for socket 0, partition 0.

After these partitions get created, we should erase and format the flash using the flashctl utility.

The commands are as follows:

```
flashctl –p /dev/fs0 –o 10M –l 1M –ev
```

This command erases the nor flash starting from an offset of 10MB to 11MB. Though the NOR flash is 128MB, the even numbered banks are 32MB each. We erase only 1MB so as to avoid overwriting the U-Boot, FM uCode, and RCW images. So for test, we erase 1MB of flash and mount a partition "flash" on that 1MB.

After giving the above command, slay the driver using the `slay devf-generic` command and then restart it again.

The Nor flash(1MB) is erased now and we can mount the given flash filesystem partition as the filesystem mountpoint /flash using the command below:

```
flashctl –p /dev/fs0p0 –o 10M –l 1M –f –n /flash
```

After this command is successfully run, slay the driver and restart it again. This formats the nor flash and mounts it over the filesystem mountpoint /flash.

We can create multiple filesystem partitions based on our requirement.

Network

This BSP includes a driver for an Intel e1000 PCIe network card as well as the Freescale P40xx DPAA network blocks. The e1000 PCIe NIC is supported by U-Boot and this BSP as a fallback interface when debugging DPAA code and configurations. The p40xx driver uses the Freescale NetComm API for talking to the DPAA blocks. This BSP is based on the NetComm v4.1 GA release. The NetCommSw release is then patched for use with the p40xx driver using the NetCommSw-P04.01.00.patch file. The original GA 4.1 release source can be downloaded from http://www.freescale.com/webapp/sps/download/mod_download.jsp?colCode=NCSW41.

Freescale's NetComm Device Drivers package supports many of Freescale's network processors. This BSP includes a QNX port of the NetComm package. The network driver, devnp-p40xx.so, uses the low level driver APIs exported by NetComm to use the DPAA blocks, specifically, the Queue Manager, Buffer Manager and the Frame Manager. The QNX port of the NetComm driver is delivered as a static library, **libNetCommSwS.a**, with the source code. This is provided separately so that the NetComm API can be used standalone by customer applications. This library is located at lib/3rdparty/netcomm. A knowledge of the architecture of the DPAA hardware blocks is necessary to understand the NetComm APIs.

The NetComm API is wrapped inside an API called the system api, (**libdpaaS.a**). This API is designed towards the needs of the io-pkt driver. It serves two purposes, one, as an integration layer between the io-pkt framework and the NetComm API, and two, an example of how the NetComm API can be used on QNX. The system API is used by devnp-p40xx.so.

The NetComm driver has extra APIs not used by the system layer. For advanced DPAA features, not currently used by the io-pkt driver, two approaches are recommended.

1. Implement new APIs in the system layer and make relevant calls to NetComm in the system layer
2. Directly call the NetComm APIs.

It is **NOT** recommended that the system layer API and the NetComm API be used simultaneously as context information is stored within the system layer.

This driver has been developed and tested for the P4080DS (Expedition) Board with the following configurations:

- SRDS_PRCTL 0x0e: SGMII card (4 x 1G) in slot 3, and 2 XAUI (10G) cards in slots 4 and 5 (only last 2 ports of 4x1G supported by hardware)
- SRDS_PRCTL 0x08: 2 XAUI (10G) cards in slots 4 and 5
- SRDS_PRCTL 0x10: SGMII card (4 x 1G) in slot 3

The network driver supports either 10 Gig interfaces on each FM or 1 Gig interfaces on each FM. The RGMII PHY interface connected on the motherboard is currently not supported in this release.

The autodetect feature of the driver shall try to detect the configuration as per the RCW settings. Users can restrict this board configuration using command line parameters. If an interface is disabled in the RCW either by the SRDS_PRCRTL setting or LPD (Lane Power Down) configuration then manual configuration through command line parameters cannot enable it. A warning will be reported in the system log when this is attempted. For the 4x1g interfaces, a warning is only reported when no interfaces are enabled. If at least one of the four is enabled, it is considered a success.

BMAN: Buffer Manager has been initialized and used for RX memory pool mainly. Although TX buffers are allocated, they are not currently being used while transmitting, because io-pkt allocates mbuf in transmit direction. BMAN Dequeue and Enqueue operations have been tested. BMAN portal and CCSR error interrupts are currently being handled. BMAN SW portal is programmed at fixed location in LAW i.e. 0xFF4000000

QMAN: Queue Manager Frame Queue Initialization, Enqueue and Dequeue operations are working. QMAN portal currently are programmed in PUSH mode only with no support of volatile dequeue commands. Currently API supports upto 1 descriptor dequeue and enqueue, which is supposed to be enhanced later. QMAN SW portal

is programmed at fixed location in LAW i.e. 0xFF4200000.

FMAN: Both FMAN1 and FMAN2 can be initialized.

PCD: This release supports PCD funtionality in FMAN. Only offline configuration of FMC tool is supported. User has to obtain the PCD tool from freescale, with which a C file can be generated. Currently, the following per-port configuration is supported for PCD (see file src/hardware/devnp/lib/dpaa/fman/fmc_config.c):

Each enabled port will distribute incoming packet streams into 8 Rx QIDs starting at PortalBase, based upon hashing of their source IP address fields. Those packets which do not have IP header shall be received into a default QID of PortalBase+8. For the 10G FM1 port, the FQID used in this scheme are 1 to 9, all belonging to Portal 4.

This PCD configuration also enables support for RX checksum offload for each configured port.

Limitations:

1. Fixed Portal to MAC Interface mapping
2. 10 Gig Optical connection is not supported on the XAUI adapter

Port Configurations

In the case of a 2 X 10Gig combination, Portal 4 is assigned to the FM1 10 Gig and Portal 9 is assigned to the FM2 10gig port. This corresponds to slots 5 and 4 respectively which are assigned interfaces tgec0 and tgec1 respectively.

In the case of a 2 X 4x1Gig (SGMII) combination in slots 5 and 4 respectively, Portals 0 to 3 and 5 to 8 are sequentially assigned to MAC 0 to 3 of FM1 and then MAC 0 to 3 of FM2. Note that this configuration is not supported by the p4080v2 silicon.

In the case of a single 4x1Gig (SGMII) card in slot 3, Portals 5 to 8 are sequentially assigned to MAC 0 to 3 of FM2 which are assigned interfaces dsec0-3.

In the 2x1Gig (SGMII) case only the last 2 ports of the 4x1Gig SGMII card are used corresponding to Portals 7 to 8, MAC 2 to 3 of FM2, and interfaces dsec0 and dsec1.

Please note that RGMII port on either FM1 or FM2 is not supported.

The SerDes must be initialized appropriately before running the driver. You can use up to four 1G ports and/or one 10G port per FMan block for a maximum total bandwidth of 12GBps/FMan.

Network Operation

Make sure that RCW settings are correct and U-Boot detects the PHYs. Here is the example output from U-Boot which shows a valid setting for 2x1G + 2x10G with an RGMII port on FM1 DTSEC2:

```
......
Net:   Fman: Uploading microcode version 101.6.0.
Dtsec: PHY is Vitesse VSC8244 (fc6c2)
Fman: Uploading microcode version 101.6.0.
Dtsec: PHY is Vitesse VSC8234 (fc623)
Dtsec: PHY is Vitesse VSC8234 (fc623)
FM1@DTSEC2, FM1@TGEC1, FM2@DTSEC3, FM2@DTSEC4, FM2@TGEC1
```

The VSC8244 PHY is for the RGMII port, the other two are for the DTSEC3-DTSEC4 on the SGMII card in slot 3. The RGMII port can also be configured as FM2 DTSEC1 in the RCW.

To start the network driver on the P4080DS board, run

```
io-pkt-v4-hc -ptcpip stacksize=65536 -dp40xx mac=00049F02XXX0
```

Replacing XXX with the last 3 digits of the P4080DS serial number. You could also not specify mac=, in which case a random mac addr is generated with the same 00049F prefix. This should create 4 interfaces corresponding to each of the 2 physical ports on SGMII card and the 2 XAUI ports, namely dsec0-dsec1 and tgec0-tgec1 respectively. Next, assign an IP address to an interface and bring it up:

```
ifconfig dsec0 192.168.255.100/24 up
```

The `/bin/netsetup` script is designed to simplify startup of the P4080DS network interfaces using DHCP to get addresses. The `/bin/netsetup` script will start the e1000 and p40xx drivers, also starting dhcp.client for the configured interfaces. It also implements a workaround for the errata resulting in the TGEC interfaces loosing the first few packets by sending packets out each interface as part of the initialization. The script usage is:

```
/bin/netsetup {auto|<p40xx-iface-select>} [p40xx-options...]
```

with the standard usage being:

```
/bin/netsetup auto mac=00049F02XXX0
```

where the XXX is replaced with the last 3 digits of the P4080DS board serial number to create a unique MAC address.

Note that the devnp-p40xx driver will configure interfaces for all ports that are allowed by the RCW configuration, it does not detect if an allowed interface card is actually plugged in. So even without the 4x1G SGMII card or the 2 XAUI cards plugged in, the driver will still report the above 4 interfaces. The interfaces without attached SGMII or XAUI cards will of course not be able to pass traffic.

Since U-Boot doesn't support tftp using the 10Gig ports, we need to use either the RGMII port on the back of the P4080DS or an Intel PCIe e1000 network card to get the QNX image via tftp. Both of these interfaces are supported by U-Boot. The e1000 card is also supported in QNX, but the RGMII port only works in U-Boot.

Network Performance Limitations

The Neutrino network stack is not designed to take full advantage of high performance hardware such as the DPAA in the p4080. Due to single threading limitations in the network stack transmit path and the Neutrino 6.5.0 SMP implementation, the devnp-p40xx network driver cannot take full advantage of the parallel bandwidth available from the DPAA networking hardware.

To take full advantage of the DPAA hardware performance requires a custom network stack that interfaces directly with the NetCommSw driver library and uses a shared memory interface to avoid memory copies between the stack and network applications.

8. Flash Bank Selection

On the P4080, the SerDes is configured by the RCW (Reset Control Word). The processor loads the RCW from flash at boot-time before starting the bootloader. The P4080DS development board has 8 flash banks and the boot bank can be configured using switch SW7[0:3]. Please read the board documentation for more information about this. Each bank needs to have an RCW, FM Microcode and uBoot image loaded to boot. Because of the mapping of the RCW and FM Microcode images only the even numbered banks are usable (the odd numbered bank RCW images overlap FM Microcode images in other even numbered banks). This also assumes no Linux images are installed in the Flash. If these are to be preserved then only banks 0 and 4 are usable.

Depending upon current bank selection, the location of U-Boot, RCW and microcode would vary. It is absolutely necessary to program the flash keeping in mind the bank switching, otherwise the programming would affect the wrong physical location in the Flash device. Flash must always be programmed with Bank 0 selected in the SW7 switches. So ideally bank 0 should never be programmed and should be kept as a backup.

The following table summarizes the location of images in U-Boot as per SW7 switch bank setting. It is assumed that programming is being done while setting flash bank to 0. Note that these are 32-bit addresses as seen in U-Boot. The actual physical address of the flash is at 0xF_E800_0000 but U-boot maps this to 0xE800_0000 while

it is running.

Bank	RCW	U-boot	Microcode
0	0xE8000000	0xEFF80000	0xEF000000
2	0xEA000000	0xEDF80000	0xED000000
4	0xEC000000	0xEBF80000	0xEB000000
6	0xEE000000	0xE9F80000	0xE9000000

9. Reset Configuration Word Settings

RCW SerDes configuration for 2x1G (slot 3) + 2x10G (slots 4 & 5)

Here are the steps to configure the SerDes for using the 10G XAUI cards on both FMs in slots 4 and 5 as well as 2 of the 1G ports on a 4x1G card in slot 3. The configuration looks like the following:

- SRDS_PRCRTL = 0x0e
    - slot1/PCIe1, slot2/PCIe3, slot3/SGMII(fm2), slot4/XAUI0(fm2), slot5/XAUI1(fm1)
- RGMII on FM1 DTSEC2
- This powers off SERDES lanes (A-D) for banks 2 & 3 and bank1 PCIe2 lanes (E-F) in the RCW
    - U-boot powers banks 2 & 3 up using SERDES8 errata rules and only the last 2 ports of SGMII are supported by this config
- Clocks: 1500Mhz CPU, 1200 MT/s DDR3, 600Mhz platform, 600 Mhz PME, 600Mhz FM

IMPORTANT The instructions are for programming the RCW in Bank4. It is assumed that the board is already able to boot into Bank 4. Before following these steps, **ensure that the board is booted up using Bank-0**. Please double check that you are using the correct values and addresses or you may render the board un-bootable.

```
mw.l 0x00100000 0xaa55aa55
mw.l 0x00100004 0x010e0100
mw.l 0x00100008 0x0c580000
mw.l 0x0010000c 0x00000000
mw.l 0x00100010 0x1e18181e
mw.l 0x00100014 0x0000cccc
mw.l 0x00100018 0x3842440c
mw.l 0x0010001c 0x3c3c2000
mw.l 0x00100020 0xfe800000
mw.l 0x00100024 0xe1000000
mw.l 0x00100028 0x00000000
mw.l 0x0010002c 0x00000000
mw.l 0x00100030 0x00000000
mw.l 0x00100034 0x008b0000
mw.l 0x00100038 0x00000000
mw.l 0x0010003c 0x00000000
mw.l 0x00100040 0x00000000
mw.l 0x00100044 0x00000000
mw.l 0x00100048 0x08138040
mw.l 0x0010004c 0xc60eec0d

protect off all
erase 0xec000000 +50
cp.b 100000 0xec000000 50

md.l ec000000 14
ec000000: aa55aa55 010e0100 0c580000 00000000    .U.U.....X......
ec000010: 1e18181e 0000cccc 3842440c 3c3c2000    ........8BD.<< .
ec000020: fe800000 e1000000 00000000 00000000    ................
ec000030: 00000000 008b0000 00000000 00000000    ................
ec000040: 00000000 00000000 08138040 c60eec0d    ...........@....
```

Change the SW switch selection to 4 now and boot using the new RCW.

Notes: tgec0 and tgec1 work except for link status (always up),
dsec0-1 work

RCW SerDes configuration for 2x10G (slots 4 & 5)

Here are the steps to configure the SerDes for using the 10G XAUI cards on both FMs in slots 4 and 5. The configuration looks like the following:

- SRDS_PRCRTL = 0x08
  - slot1/PCIe1, slot2/PCIe3, slot3/PCIe2, slot4/XAUI0(fm2), slot5/XAUI1(fm1)
- RGMII on FM1 DTSEC2
- This powers off SERDES lanes for banks 2 & 3 in the RCW
  - U-boot powers up banks 2 & 3 using SERDES8 errata rules
- Clocks: 1500Mhz CPU, 1200 MT/s DDR3, 600Mhz platform, 600 Mhz PME, 600Mhz FM

IMPORTANT The instructions are for programming the RCW in Bank4. It is assumed that the board is already able to boot into Bank 4. Before following these steps, **ensure that the board is booted up using Bank-0**. Please double check that you are using the correct values and addresses or you may render the board un-bootable.

```
mw.l 0x00100000 0xaa55aa55
mw.l 0x00100004 0x010e0100
mw.l 0x00100008 0x4c580000
mw.l 0x0010000c 0x00000000
mw.l 0x00100010 0x1e18181e
mw.l 0x00100014 0x0000cccc
mw.l 0x00100018 0x20404400
mw.l 0x0010001c 0x3c3c2000
mw.l 0x00100020 0xfe800000
mw.l 0x00100024 0xe1000000
mw.l 0x00100028 0x00000000
mw.l 0x0010002c 0x00000000
mw.l 0x00100030 0x00000000
mw.l 0x00100034 0x008b0000
mw.l 0x00100038 0x00000000
mw.l 0x0010003c 0x00000000
mw.l 0x00100040 0x00000000
mw.l 0x00100044 0x00000000
mw.l 0x00100048 0x08138040
mw.l 0x0010004c 0x4e684bb1

protect off all
erase 0xec000000 +50
cp.b 100000 0xec000000 50

md.l ec000000 14
ec000000: aa55aa55 010e0100 4c580000 00000000    .U.U....LX......
ec000010: 1e18181e 0000cccc 20404400 3c3c2000    ........ @D.<< .
ec000020: fe800000 e1000000 00000000 00000000    ................
ec000030: 00000000 008b0000 00000000 00000000    ................
ec000040: 00000000 00000000 08138040 4e684bb1    ...........@NhK.
```

Change the SW switch selection to 4 now and boot using the new RCW.

Notes: tgec0 and tgec1 work except for link status (always up)

RCW SerDes configuration for 4x1G (slot 3)

Here are the steps to configure the SerDes for using an SGMII card in slot3 with FM2 The configuration looks like the following:

- SRDS_PRCRTL = 0x10
  - slot1/PCIe1, slot2/PCIe3, slot3/SGMII(fm2)
- RGMII on FM1 DTSEC2
- This powers off SERDES lanes for banks 2 & 3 in the RCW
  - **U-boot requires `hwconfig=serdes:fsl_srds_lpd_b2=0xf` env setting to disable bank 2 SERDES power**
  - The bank 3 SERDES is not powered up by U-Boot in this config
- Clocks: 1500Mhz CPU, 1200 MT/s DDR3, 600Mhz platform, 600 Mhz PME, 600Mhz FM

IMPORTANT The instructions are for programming the RCW in Bank4. It is assumed that the board is already able to boot into Bank 4. Before following these steps, **ensure that the board is booted up using Bank-0**. Please double check that you are using the correct values and addresses or you may render the board un-bootable.

```
mw.l 0x00100000 0xaa55aa55
mw.l 0x00100004 0x010e0100
mw.l 0x00100008 0x4c580000
mw.l 0x0010000c 0x00000000
mw.l 0x00100010 0x1e18181e
mw.l 0x00100014 0x0000cccc
mw.l 0x00100018 0x40464400
mw.l 0x0010001c 0x3c3c2000
mw.l 0x00100020 0xfe800000
mw.l 0x00100024 0xe1000000
mw.l 0x00100028 0x00000000
mw.l 0x0010002c 0x00000000
mw.l 0x00100030 0x00000000
mw.l 0x00100034 0x008b0000
mw.l 0x00100038 0x00000000
mw.l 0x0010003c 0x00000000
mw.l 0x00100040 0x00000000
mw.l 0x00100044 0x00000000
mw.l 0x00100048 0x08138040
mw.l 0x0010004c 0xe0388c7a

protect off all
erase 0xec000000 +50
cp.b 100000 0xec000000 50

md.l ec000000 14
ec000000: aa55aa55 010e0100 4c580000 00000000    .U.U....LX......
ec000010: 1e18181e 0000cccc 40464400 3c3c2000    ........@FD.<< .
ec000020: fe800000 e1000000 00000000 00000000    ................
ec000030: 00000000 008b0000 00000000 00000000    ................
ec000040: 00000000 00000000 08138040 e0388c7a    ...........@.8.z
```

Change the SW switch selection to 4 now and boot into U-Boot using the new RCW.

```
# Now disable the bank2 ~SerDes lanes since they are unusable in this config.
# Note: do the following after booting into bank 4, not in bank 0!!
setenv hwconfig $hwconfig_orig\;serdes:fsl_srds_lpd_b2=0xf
```

Notes: dsec0-3 work

RCW SerDes configuration for 4x1G (slot 3) Alternative Clocking

Here are some additional clocking configurations for the 4x1G (slot 3) setup. Follow the instructions from above but using the following RCW words.

The first configuration looks like the following:

- SRDS_PRCRTL = 0x10
    - slot1/PCIe1, slot2/PCIe3, slot3/SGMII(fm2)
- RGMII on FM1 DTSEC2
- This powers off SERDES lanes for banks 2 & 3 in the RCW
    - **U-boot requires `hwconfig=serdes:fsl_srds_lpd_b2=0xf` env setting to disable bank 2 SERDES power**
    - The bank 3 SERDES is not powered up by U-Boot in this config
- Clocks: 1500Mhz CPU, 1300 MT/s DDR3, 800Mhz platform, 600 Mhz PME, 600Mhz FM

```
mw.l 0x00100000 0xaa55aa55
mw.l 0x00100004 0x010e0100
mw.l 0x00100008 0x105a0000
mw.l 0x0010000c 0x00000000
mw.l 0x00100010 0x1e18181e
mw.l 0x00100014 0x0000cccc
mw.l 0x00100018 0x40464400
mw.l 0x0010001c 0x3c3c2000
mw.l 0x00100020 0xfe800000
mw.l 0x00100024 0xe1000000
mw.l 0x00100028 0x00000000
mw.l 0x0010002c 0x00000000
mw.l 0x00100030 0x00000000
mw.l 0x00100034 0x008b0000
mw.l 0x00100038 0x00000000
mw.l 0x0010003c 0x00000000
mw.l 0x00100040 0x00000000
mw.l 0x00100044 0x00000000
mw.l 0x00100048 0x08138040
mw.l 0x0010004c 0x1397796e
```

The second configuration is for a 133.33Mhz system clock. To get this you'll have to change SW3 6-8 from its default of 100 (100Mhz) to 111 (133.333Mhz). You will also have to update the build file startup line '-t' option to set the correct system clock speed:

```
    [+keeplinked]startup-p4080ds -c0xFFE000000 -t133333000 -v
```

The resulting configuration looks like the following:

- SRDS_PRCRTL = 0x10
    - slot1/PCIe1, slot2/PCIe3, slot3/SGMII(fm2)
- RGMII on FM1 DTSEC2
- This powers off SERDES lanes for banks 2 & 3 in the RCW
    - **U-boot requires `hwconfig=serdes:fsl_srds_lpd_b2=0xf` env setting to disable bank 2 SERDES power**
    - The bank 3 SERDES is not powered up by U-Boot in this config
- Clocks: 1467Mhz CPU, 1333 MT/s DDR3, 800Mhz platform, 600 Mhz PME, 600Mhz FM

```
mw.l 0x00100000 0xaa55aa55
mw.l 0x00100004 0x010e0100
mw.l 0x00100008 0x0c540000
mw.l 0x0010000c 0x00000000
mw.l 0x00100010 0x16121216
mw.l 0x00100014 0x0000cccc
mw.l 0x00100018 0x40464400
mw.l 0x0010001c 0x3c3c2000
mw.l 0x00100020 0xfe800000
mw.l 0x00100024 0xe1000000
mw.l 0x00100028 0x00000000
mw.l 0x0010002c 0x00000000
mw.l 0x00100030 0x00000000
mw.l 0x00100034 0x008b0000
mw.l 0x00100038 0x00000000
mw.l 0x0010003c 0x00000000
mw.l 0x00100040 0x00000000
mw.l 0x00100044 0x00000000
mw.l 0x00100048 0x08138040
mw.l 0x0010004c 0x825ebe28
```

10. Known Issues

10.1 Fixed in this Release

4. The 36-bit IMMR/CCSR region at 0xF_FE00_0000 configured by U-Boot is moved by this BSP to under 4Gb at 0x0_FE00_0000 to avoid a multi-processor startup hanging issue. Once this issue is corrected the CCSR/IMMR will be reverted to its 36-bit value.

6. Approx 20 packets are lost from the 10Gig ports (tgec0 and tgec1) in transmit direction after initialization is complete. This is a know issue and as a work around, a ping traffic should be generated from each of the interface initially. Any test should only be run once ping to external interface is successful. This is a hardware issue and will be fixed in the next rev of the silicon.

8. Numerous "`Cannot find paddr for vaddr`" error messages are reported in sloginfo. These do not indicate a problem and the messages will be removed in a later release.

9. On devnp-p40xx driver startup the message "unknown relocation type in devnp-p40xx.so. Symbol: __start" is reported. This does not indicate a problem and the message will removed in a later release.

10. The PHY for the last interface (DTSEC4) on the 4x1g SGMII card does not report link status correctly. The interface transfers packets but looses ~1% packets and reports the link as down. This affects dsec1 in a 2x1g configuration.

16. The pci-p4080 server reports "not enough memory" errors in sloginfo. This does not affect PCIe device functioning.

18. Interleaved memory configurations are not supported by this BSP

19. PCIe cards placed in slot 3 (PCIe2) don't work in an RCW config with PCIe2 enabled

20. Although 16Gb of memory can be installed (2x8Gb), only the first 4Gb of memory is currently usable by Neutrino (minus the 16Mb used by the CCSR/IMMR region).

21. The interfaces can start to fail with "`Failed to Enqueue Packet. Error 0xd0020`" errors when pushed. This can happen at data rates as low as 60MBytes/s. These errors can seriously impact performance. The problem is intermittent and can sometimes be mitigated by reducing packet sizes. This will be corrected in a later release.

22. FTP and NFS/UDP transfers of large files are seeing some data corruption in the transfered file. We believe this is a hardware specific memory issue and are investigating the issue further.

23. Interfaces can fail under high packet receive load with various "`mbuf signature`" and "`Cannot find vaddr for paddr 0`" errors.

24. Low transaction rates (<2000/s) seen with netperf TCP_RR test on 1G and 10G interfaces. With p4080 talking to an Intel 8-core linux server we now get >9000/s, talking to a mpc8548 using io-net driver we get ~8600/s, but talking to another p4080 we get ~7600/s. Better, but needs more investigation.

26. "`Failed to Enqueue Packet. Error 0xd0020`" error messages seen in sloginfo output during netperf TCP_STREAM transmit test. These are generally just warnings as an automatic retry corrects the issue.

27. netperf is occasionally producing "`netperf: data send error: Socket operation on non-socket`" and "`len was -1`" error messages. They do not appear to affect the tests but need further investigation.

29. jumbo packets not supported by devnp-p40xx driver

30. The ehci-p2020 USB driver doesn't handle memory above 4Gb therefore it is disabled in the default build file which uses the procnto -mP switch to allocate from memory above 4Gb by default. You can enable the USB driver on a machine with <=4Gb of memory or if you remove the -mP option on a machine with >4Gb of physical memory (in which case only 4Gb is usable). This will be fixed in a later release.

31. The p4080 can get into a state where a high rate of #40 interrupts are generated (>150k/s). When this happens the message "`p40xx_process_common_err_intr: High err intr count 1500000, error interrupt 0x4`" is logged after ~10 seconds. This is an 'Internal RAM Multi-bit ECC Error' and when it occurs you'll need to reboot.
33. Drivers for I2C interfaces 1 & 2 cannot be enabled at the same time. When the second driver is started only one of them works.
34. When transmitting at high load in parallel across multiple interfaces one or more of the interfaces can stop transmitting. Once in this state the affected interfaces can only be restarted by rebooting the machine. This is QNX Ref# 94815.
35. Interrupts IntB-D for mutli-function PCIe devices are not configured.
36. promiscuous and multi-cast support is not implemented in devnp-p40xx driver.
37. lshwi reports incorrect data for 3 PCIe buses
38. The SPI interface is not supported in Neutrino
42. can run out of MBUFs under high network load.
43. using devnp-p40xx command line options to set network speed and duplex fails to work.
44. PCIe DMA to SDRAM with addresses overlapping the PCIe memory window for that bus silently fails to work.

10.2 Remaining Issues

1. The P4080DS board does not have a NAND flash device installed.
2. The P4080DS board does not have a Promjet device installed.
3. The network driver cannot be restarted. This is because the buffer manager block cannot be re-seeded. See QNX Ref# 73621 for details.
5. SGMII PHY connected at FM2 are not accessible without muxing GPIO pins and that reduces performance of PHY polling. It is recommended to use emu_phy option in case both FM1 and FM2 Phys are being managed.
7. There are various errata and SW workaround implemented in network driver for particular revision of the chip. Refer to the latest documents from Freescale to understand the impact on SW and whether the workaround would apply to your system or not.
11. The DPAA will configure calculated random MAC addrs on its interfaces unless the **mac=** option is used to configure a starting MAC address for the 10 possible MAC interfaces. To get a unique MAC we suggest you use **`mac=00049f02###0`**, replacing `###` with the last 3 digits of the board serial number. You need to ensure that this number is unique on you local network. You should also set ethaddr, eth1addr, ..., eth7addr in U-Boot starting with this MAC for ethaddr and adding 1 for each of eth1addr to eth7addr to the last digit of the MAC.
12. The 10G interfaces use an emulated PHY that always shows the interface as up. They cannot indicate actual PHY connectivity status.
13. The 10G interfaces report their media as 10GbaseCX4 instead of 10GbaseT. This will be corrected in a later SDP and BSP release.
14. The network startup will sometimes hang at "Waiting for address from DHCP..." after a system power up. A system reset is required to correct the problem. A normal startup will pause for a few seconds at this point, so wait at least 30s-60s before considering it hung.

15. The DPAA network interfaces can sometimes take a while to get addresses with DHCP. They do eventually get their addresses but you may see the interfaces up but without an address just after booting. It may sometimes take up to 30-90s to get an address.

17. The SD/MMC card is not supported in Neutrino (SD/MMC is supported in U-boot).

25. `"Processing EISR0_INT_SOURCE_FM2"` messages seen in sloginfo output during netperf TCP_STREAM transmit test. This generally is due to packet counter overflows which can safely be ignored.

28. The system is occasionally hanging in U-Boot after the memory config report. This appears to be hardware and/or memory DIMM type specific. A reset corrects the problem.

32. There are 19 linking warnings when built the source code src/hardware/startup/boards/p4080ds. It looks like the output below. You can safely ignore these warnings. (Ref# 74046)
:C:\QNX650\host\win32\x86\usr\bin\ntoppc-ld: Warning:
C:/QNX650/target/qnx6/ppcbe/usr/lib\libucl.a(n2b_d.o) uses hard float,
src/hardware/startup/boards/p4080ds/ppc/be/startup-p4080ds uses soft float.

39. The NetCommSw FM_10G_TX_ECC_FRMS_ERRATA_10GMAC_A004 errata fix is disabled. When 2x10G interfaces are configured it causes the second interface to fail to initialize with an error saying that it is already enabled.

40. Network performance across interfaces run in parallel does not scale with the number of interfaces (limit is currently ~1.03Gb/s total across 4x1Gb interfaces).

41. Network performance on the 10G interface is limited to ~2Gb/s (and this requires the use of 9k jumbo packets).

45. On some of the boards where both 4x1G SGMII slots are being used, driver may abruptly exit because of no PLL lock. Following log shall be seen in the sloginfo output

    ```
    Jan 01 00:00:12    5 18765     0 ! MINOR ??? Error
    [/home/agoswami/src/trunk/rep/src/hardware/devnp/lib/dpaa/system/platform_p4080_ds.c:509
    PlatformSetSerdesErratas]: Invalid State;
    Jan 01 00:00:12    5 18765     0 Bank 2 : no PLL lock
    ```

    At this point the board needs to be restarted with a power cycle.